



SPYWOLF

Security Audit Report

CONTRACTS ARE DEPLOYED ON TESTNET



Completed on
July 25, 2022

MADE IN USA 

@SPYWOLFNETWORK



@SPYWOLFNETWORK



SPYWOLF.CO





OVERVIEW

This audit has been prepared for **Fortunas Finance** to review the main aspects of the project to help investors make an informative decision during their research process.

You will find a summarized review of the following key points:

- ✓ Contract's source code
- ✓ Owners' wallets
- ✓ Tokenomics
- ✓ Team transparency and goals
- ✓ Website's age, code, security and UX
- ✓ Whitepaper and roadmap
- ✓ Social media & online presence

“

The results of this audit are purely based on the team's evaluation and does not guarantee nor reflect the projects outcome and goal

- SPYWOLF Team -

”





TABLE OF CONTENTS

Project Description	01
Contract 1 – Stats	02–09
Contract 2 – Stats	10–12
Contract 3 – Stats	13–15
Tokenomics	16
Team Information	17
Website Analysis	18
Social Media & Online Presence	19
About SPYWOLF	20
Disclaimer	21



Fortunas Finance



PROJECT DESCRIPTION

According to their website, Fortunas Finance is building a smart Defi ecosystem, one that has a fundamental grasp on the principle of supply and demand. Gone are the days of Ponzinomics, every Fortunas Finance investor requires contributing resources in order to generate cash flow. By using a smart risk-reward structure, we are building an exciting protocol that will reward holders for years to come. All while building multiple revenue streams and garnering investor trust.

Release Date: Presale starts on July, 2021

Category: Rewards/APY



CONTRACT INFO

Token Name

Fortunas Token

Symbol

FRTNA

Contract Address

0x97A3f281CC8c32AE570a711576C3ba5Ffd818F0B

Network

Rinkeby **TESTNET**

Language

Solidity

Deployment Date

July 25, 2022

Verified?

Yes

Total Supply

500,000,000

Status

Deployed

TAXES

Buy Tax
10%

Sell Tax
10%

*Taxes can be changed in future



Our Contract Review Process

The contract review process pays special attention to the following:

- ✓ Testing the smart contracts against both common and uncommon vulnerabilities
- ✓ Assessing the codebase to ensure compliance with current best practices and industry standards.
- ✓ Ensuring contract logic meets the specifications and intentions of the client.
- ✓ Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- ✓ Thorough line-by-line manual review of the entire codebase by industry experts.

Blockchain security tools used:

- OpenZeppelin
- Mythril
- Solidity Compiler
- Hardhat



CURRENT STATS

(As of July 22, 2022)



Liquidity

Not added yet



Burn

No burnt tokens

Status:
Not Launched!

MaxTxAmount
No limit

DEX:
Unknown

LP Address(es)

Liquidity not added yet



TOKEN TRANSFERS STATS

Transfer Count	Token is deployed on testnet
Uniq Senders	Token is deployed on testnet
Uniq Receivers	Token is deployed on testnet
Total Amount	Token is deployed on testnet
Median Transfer Amount	Token is deployed on testnet
Average Transfer Amount	Token is deployed on testnet
First transfer date	Token is deployed on testnet
Last transfer date	Token is deployed on testnet
Days token transferred	Token is deployed on testnet

SMART CONTRACT STATS

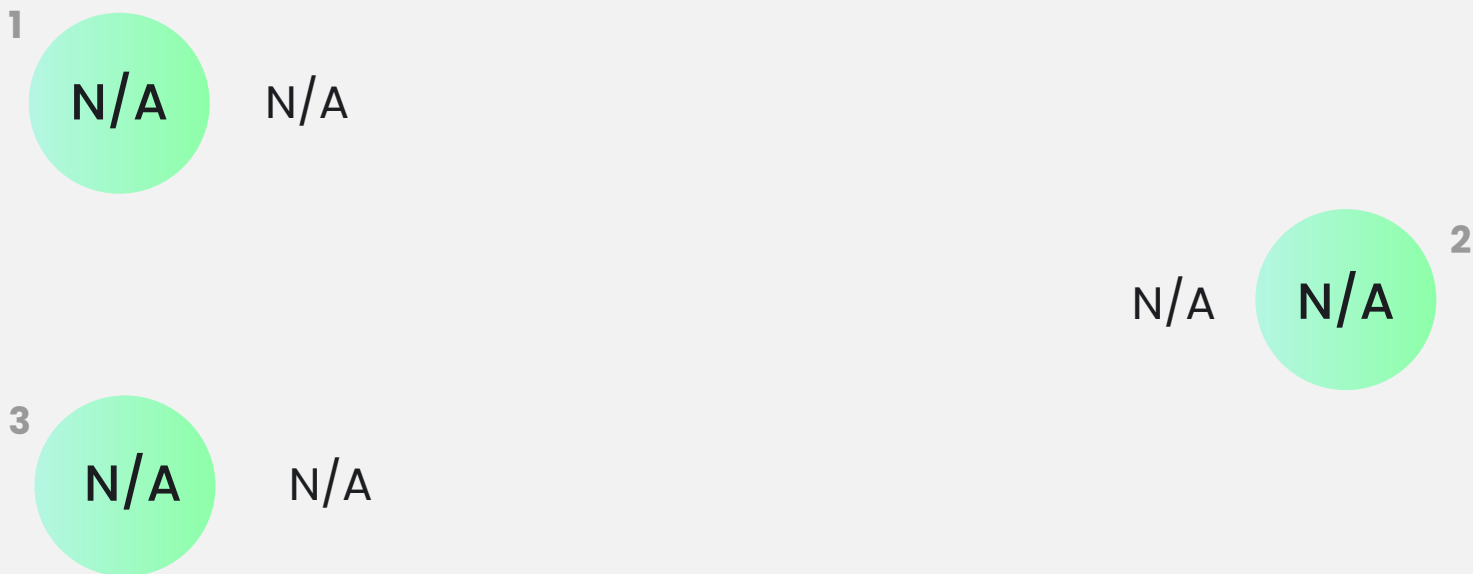
Calls Count	Token is deployed on testnet
External calls	Token is deployed on testnet
Internal calls	Token is deployed on testnet
Transactions count	Token is deployed on testnet
Uniq Callers	Token is deployed on testnet
Days contract called	Token is deployed on testnet
Last transaction time	Token is deployed on testnet
Created	Token is deployed on testnet
Create TX	Token is deployed on testnet
Creator	Token is deployed on testnet



FEATURED WALLETS

Owner address	0x45faf7923bab5a5380515e055ca700519b3e4705
Fortunas ledger	0xcb150ebbd71d96c835102b0207b95012dc730d8d
Liquidity wallet	Same as owner
Reward wallet	0x3edce801a3f1851675e68589844b1b412eac6b07
Treasury wallet	0x49a61ba8e25fbd58ce9b30e1276c4eb41dd80a80
LP address	Liquidity not added yet

TOP 3 UNLOCKED WALLETS





VULNERABILITY CHECK

Design Logic	Passed
Compiler warnings.	Passed
Private user data leaks	Passed
Timestamp dependence	Passed
Integer overflow and underflow	Passed
Race conditions and reentrancy. Cross-function race conditions	Passed
Possible delays in data delivery	Passed
Oracle calls	Passed
Front running	Passed
DoS with Revert	Passed
DoS with block gas limit	Passed
Methods execution permissions	Passed
Economy model	Passed
Impact of the exchange rate on the logic	Passed
Malicious Event log	Passed
Scoping and declarations	Passed
Uninitialized storage pointers	Passed
Arithmetic accuracy	Passed
Cross-function race conditions	Passed
Safe Zeppelin module	Passed
Fallback function security	Passed



THREAT LEVELS

When performing smart contract audits, our specialists look for known vulnerabilities as well as logical and access control issues within the code. The exploitation of these issues by malicious actors may cause serious financial damage to projects that failed to get an audit in time. We categorize these vulnerabilities by the following levels:

High Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

Medium Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

Low Risk

Issues on this level are minor details and warning that can remain unfixed.

Informational

Information level is to offer suggestions for improvement of efficacy or security for features with a risk free factor.



FOUND THREATS

⚠ High Risk

Owner can change reward wallet.

When claimLedger() function is triggered, if the rewards wallet balance is zero, rewards owed to the holder called the function will be minted (this will create and add additional tokens to the total supply).

```
function updateRewardWallet(address newRewardWallet) public onlyOwner {
    require(newRewardWallet != rewardWallet,
        "FRTNA::The staking wallet is already this address");
    excludeFromFees(rewardWallet, false);
    excludeFromFees(newRewardWallet, true);
    emit UpdatedRewardWallet(newRewardWallet, rewardWallet);
    rewardWallet = newRewardWallet;
}

function claimLedger() external {
    .....
    if (isMint) {
        if (rewardWalletBalance != 0) {
            _transfer(rewardWallet, msg.sender, rewardWalletBalance);
        }

        _mint(msg.sender, totalPassiveRewards.sub(rewardWalletBalance));
    }
    .....
}
```



FOUND THREATS

⚠ Medium Risk

Battling contract can mint new tokens and burn existing tokens.

```
modifier onlyContract {  
    require(msg.sender == battling,  
        "FRTNA::Only Fortunas Battling Contract can call this function");  
    _;  
}  
  
function mint(address account, uint256 amount) external onlyContract {  
    _mint(account, amount);  
}  
  
function burn(address account, uint256 amount) external onlyContract {  
    _burn(account, amount);  
}
```



⚠ Low Risk

Owner can set buy/sell fees up to 10% (combined buy+sell=20%).

```
uint256 public maxBuyingFee;
uint256 public maxSellingFee;
uint256 public immutable multiplierForTotalFee;

uint256 _liquidityBuyingFee = 25;
uint256 _treasuryBuyingFee = 75;
uint256 _burnBuyingFee = 0;

uint256 _liquiditySellingFee = 50;
uint256 _treasurySellingFee = 25;
uint256 _burnSellingFee = 25;

maxBuyingFee = _liquidityBuyingFee.add(_treasuryBuyingFee).add(_burnBuyingFee);
maxSellingFee = _liquiditySellingFee.add(_treasurySellingFee).add(_burnSellingFee);

function updateBuyingFees(uint256 newLiquidityBuyingFee,
    uint256 newTreasuryBuyingFee, uint256 newBurnBuyingFee) public onlyOwner {
    uint256 totalInputFee = newLiquidityBuyingFee.add(newTreasuryBuyingFee).add(newBurnBuyingFee);
    require(totalInputFee <= maxBuyingFee, "FRTNA::Cannot exceed total Buying fees");

    liquidityBuyingFee = newLiquidityBuyingFee;
    treasuryBuyingFee = newTreasuryBuyingFee;
    burnBuyingFee = newBurnBuyingFee;
}

function updateSellingFees(uint256 newLiquiditySellingFee,
    uint256 newTreasurySellingFee, uint256 newBurnSellingFee) public onlyOwner {
    uint256 totalInputFee = newLiquiditySellingFee.add(newTreasurySellingFee).add(newBurnSellingFee);
    require(totalInputFee <= maxSellingFee, "FRTNA::Cannot exceed total selling fees");

    liquiditySellingFee = newLiquiditySellingFee;
    treasurySellingFee = newTreasurySellingFee;
    burnSellingFee = newBurnSellingFee;
}
```




Informational

Owner can set battling contract only once.

```
function initializeBattling(address battlingContractAddress) external onlyOwner {
    require(battling == address(0), "FRTNA::Battling has already been initialized");
    battling = battlingContractAddress;

    excludeFromPassiveRewards(battling, true);

    excludeFromFees(battling, true);
}
```

Owner can exclude address from fees.

```
function excludeFromFees(address account, bool excluded) public onlyOwner {
    require(isExcludedFromFees[account] != excluded, "FRTNA::Account is already the value of 'excluded'");
    isExcludedFromFees[account] = excluded;

    emit ExcludedFromFees(account, excluded);
}

function excludeMultipleAccountsFromFees(address[] calldata accounts, bool excluded) public onlyOwner {
    for(uint256 i = 0; i < accounts.length; i++) {
        isExcludedFromFees[accounts[i]] = excluded;
    }

    emit ExcludedMultipleAccountsFromFees(accounts, excluded);
}
```



Informational

Owner can exclude address from passive rewards.

```
function excludeFromPassiveRewards(address account, bool excluded) public onlyOwner {
    require(isExcludedFromPassiveRewards[account] != excluded,
        "FRTNA::Account is already the value of 'excluded'");

    isExcludedFromPassiveRewards[account] = excluded;
}

function excludeMultipleAccountsFromPassiveRewards(
    address[] calldata accounts, bool excluded) public onlyOwner {
    for(uint256 i = 0; i < accounts.length; i++) {
        isExcludedFromPassiveRewards[accounts[i]] = excluded;
    }
}
```




RECOMMENDATIONS FOR

GOOD PRACTICES

1

Consider fundamental tradeoffs

2

Be attentive to blockchain properties

3

Ensure careful rollouts

4

Keep contracts simple

5

Stay up to date and track development

Fortunas Finance

GOOD PRACTICES FOUND

- ✓ The owner cannot stop or pause the contract
- ✓ The owner cannot set a transaction limit
- ✓ The smart contract utilizes "SafeMath" to prevent overflows

CONTRACT INFO

Token Name
Battling

Symbol
N/A

Contract Address

0x792755777304EA2b1D821F8Eb1bc585b8845c762

Network

Rinkeby **TESTNET**

Language

Solidity

Deployment Date

July 25, 2022

Verified?

Yes

Total Supply

N/A

Status

Deployed

TAXES

Buy Tax

N/A

Sell Tax

N/A

*There are 0.5% taxes leaving a battle



Our Contract Review Process

The contract review process pays special attention to the following:

- ✓ Testing the smart contracts against both common and uncommon vulnerabilities
- ✓ Assessing the codebase to ensure compliance with current best practices and industry standards.
- ✓ Ensuring contract logic meets the specifications and intentions of the client.
- ✓ Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- ✓ Thorough line-by-line manual review of the entire codebase by industry experts.

Blockchain security tools used:

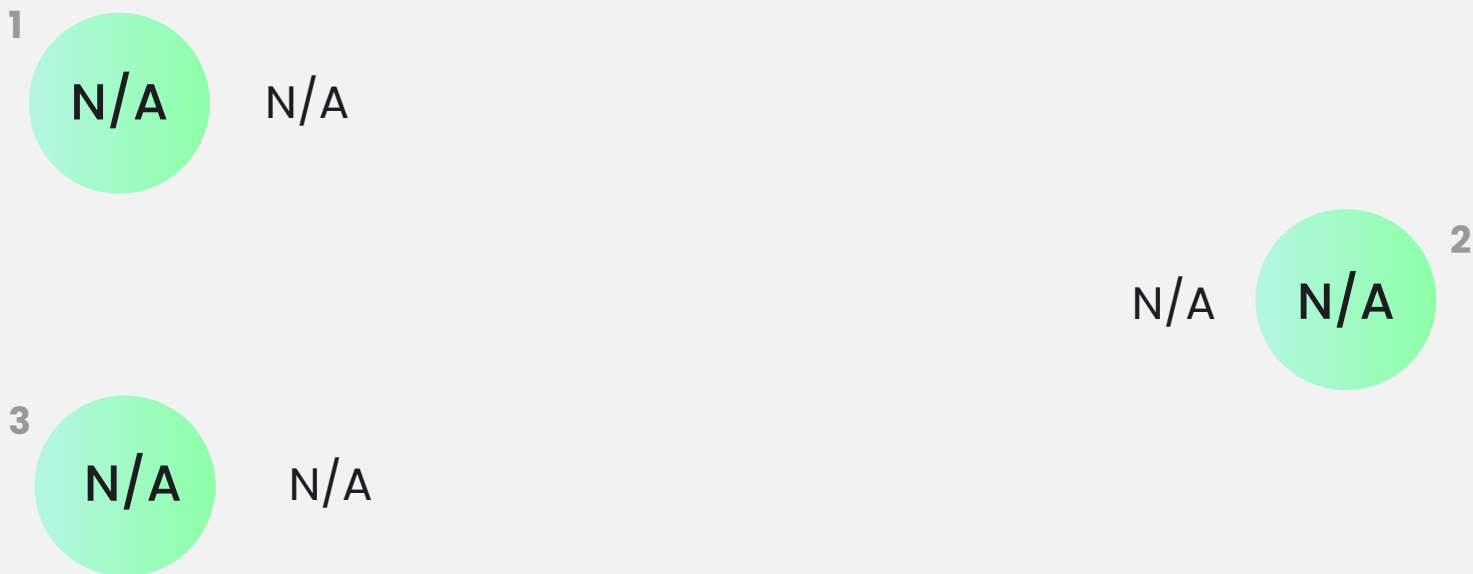
- OpenZeppelin
- Mythril
- Solidity Compiler
- Hardhat



FEATURED WALLETS

Owner address	0x45faf7923bab5a5380515e055ca700519b3e4705
Reward wallet	0x3edce801a3f1851675e68589844b1b412eac6b07
Treasury wallet	0x49a61ba8e25fbd58ce9b30e1276c4eb41dd80a80
LP address	Liquidity not added yet

TOP 3 UNLOCKED WALLETS





FOUND THREATS

⚠ Medium Risk

The endBattle() function mint new tokens in the fortunasToken's contract and distribute them as holder rewards if the rewardWallet is empty.

This can lead in token's price inflation in future.

```
function endBattle(  
uint8 _battleType  
) external {  
.....  
if (isMint) {  
    if (rewardWalletBalance != 0) {  
        fortunasToken.transferFrom(rewardWallet, msg.sender, rewardWalletBalance);  
    }  
  
    fortunasToken.mint(msg.sender, rewardsToReturn.sub(rewardWalletBalance));  
}  
else {  
    fortunasToken.transferFrom(rewardWallet, msg.sender, rewardsToReturn);  
}  
.....  
}
```



Informational

Owner can change treasury and rewards wallet address.

```
function setTreasuryWallet(address _treasuryWallet) external onlyOwner {
    require(treasuryWallet != _treasuryWallet, "setTreasuryWallet::TW");
    emit UpdatedTreasuryWallet(_treasuryWallet, treasuryWallet);
    treasuryWallet = _treasuryWallet;
}

function setRewardWallet(address _rewardWallet) external onlyOwner {
    require(rewardWallet != _rewardWallet, "setRewardWallet::RW");
    emit UpdatedRewardWallet(_rewardWallet, rewardWallet);
    rewardWallet = _rewardWallet;
}
```

Owner can change battle percentage rates.

```
function setBattleResetPercentage(uint256 _battleResetPercentage) external onlyOwner {
    require(battleResetPercentage != _battleResetPercentage, "setBattleResetPercentage::BRP");
    emit UpdatedBattleResetPercentage(_battleResetPercentage, battleResetPercentage);
    battleResetPercentage = _battleResetPercentage;
}
```

Functions with onlyOwner restriction are involved into external functions. This can lead to revert of the call.

```
modifier onlyOwner() {
    require(owner() == _msgSender(), "Ownable: caller is not the owner");
    _;
}

function createAssetRandomness() external view onlyOwner returns (uint256) {}

function calculateRewards(Battle memory _tempBattle)
    public view onlyOwner returns (Battle memory) {}

function calculateRewardsForEndBattle(Battle memory _tempBattle)
    external view onlyOwner returns (Battle memory) {}

function calculateRations(Battle memory _tempBattle, uint256 _rationDays)
    external view onlyOwner returns (uint256) {}

function createRandomness(uint256 _chance, uint256 _multiplier, uint256 _helper)
    public view onlyOwner returns (bool) {}
```

CONTRACT INFO

Token Name
FortunasAssets

Symbol
N/A

Contract Address
0x18716E8D10A9D69d502A16F0A69A66ac7Ac26220

Network
Rinkeby **TESTNET**

Language
Solidity

Deployment Date
July 25, 2022

Verified?
Yes

Total Supply
N/A

Status
Deployed

TAXES

Buy Tax
none

Sell Tax
none



Our Contract Review Process

The contract review process pays special attention to the following:

- ✓ Testing the smart contracts against both common and uncommon vulnerabilities
- ✓ Assessing the codebase to ensure compliance with current best practices and industry standards.
- ✓ Ensuring contract logic meets the specifications and intentions of the client.
- ✓ Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- ✓ Thorough line-by-line manual review of the entire codebase by industry experts.

Blockchain security tools used:

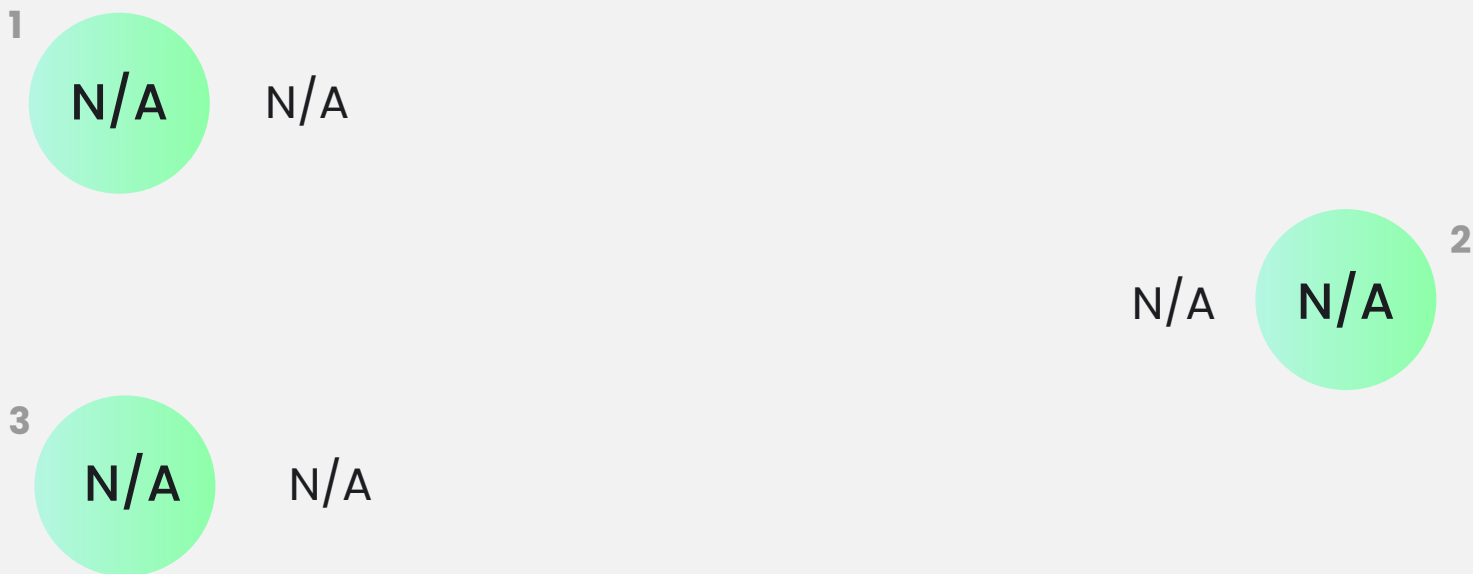
- OpenZeppelin
- Mythril
- Solidity Compiler
- Hardhat



FEATURED WALLETS

Owner address	0x45faf7923bab5a5380515e055ca700519b3e4705
Battling	0x792755777304ea2b1d821f8eb1bc585b8845c762
LP address	Liquidity not added yet

TOP 3 UNLOCKED WALLETS





FOUND THREATS

⚠ Medium Risk

Owner can disable NFT transfers.

```
function setIsTransferEnabled(  
    bool _state  
) external onlyOwner {  
    require(isTransferEnabled != _state,  
        "setIsTransferEnabled::isTransferEnabled is already set to this state");  
    isTransferEnabled = _state;  
}
```



Informational

Owner can initialize battling address once.

```
function initializeBattling(  
    address _battling  
) external onlyOwner {  
    require(battling == address(0),  
        "initializeBattling::Battling has already been initialized");  
    battling = _battling;  
}
```

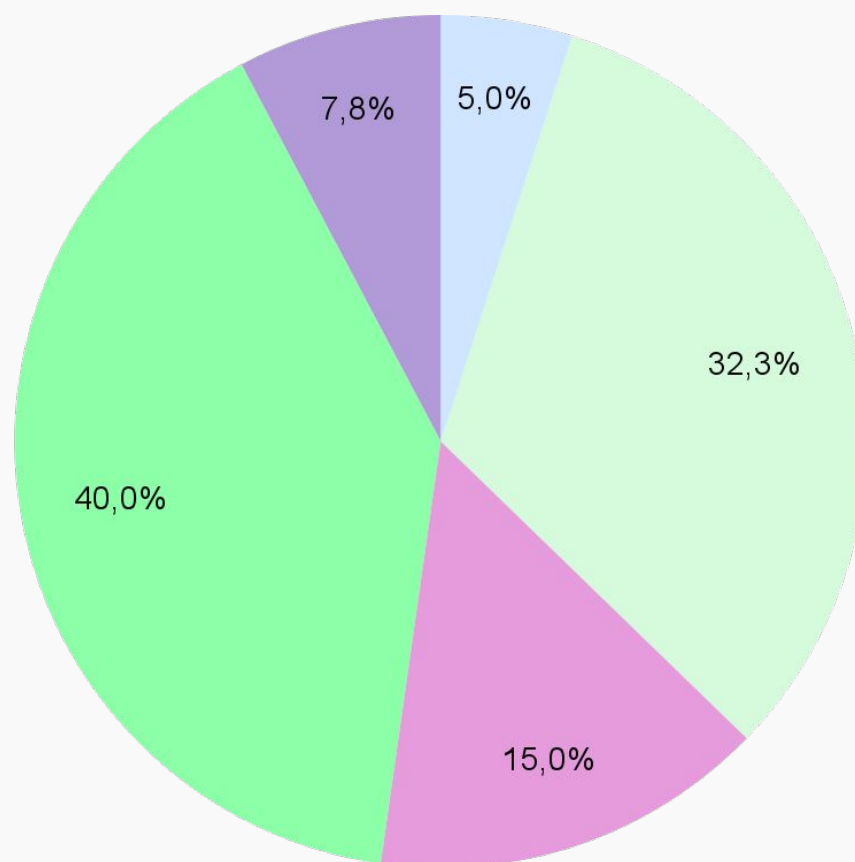
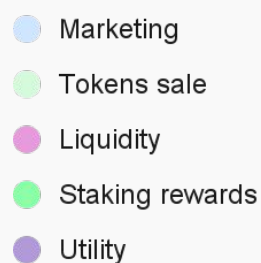
Owner can change NFTs URI (image).

```
function setURI(  
    string memory _uri  
) external onlyOwner {  
    _setURI(_uri);  
}
```



*The following tokenomics are based on the project's whitepaper and/or website:

- 32.25% - Token sales
- 15% - Liquidity
- 5% - Marketing
- 40% - Staking rewards
- 7.75% - Utility



TOKENOMICS



THE TEAM

The team has privately doxxed to SPYWOLF by completing the following KYC requirements:

- ID Verification
- Video statement
- Video interview with devs
- Owner's wallet verification

KYC INFORMATION

Issuer

SPYWOLF

Members KYC'd



KYC Date

July 25, 2022

Format

Image

Certificate Link

https://github.com/SpyWolfNetwork/KYCs/blob/main/July/KYC_Fortunas_Token_0x7759D8345badb3cBec4ff0AC0BE08Ad9BBc8A2A8.png





WEBSITE

Website URL

<https://fortunas.finance/>

Domain Registry

<https://www.godaddy.com>

Domain Expiration

Expires on 2023-04-18

Technical SEO Test

Passed

Security Test

Passed. SSL certificate present

Design

Single page design,
appropriate color scheme
and graphics.

Content

The information helps new
investors understand what
the product does right away.
No grammar mistakes
found.

Whitepaper

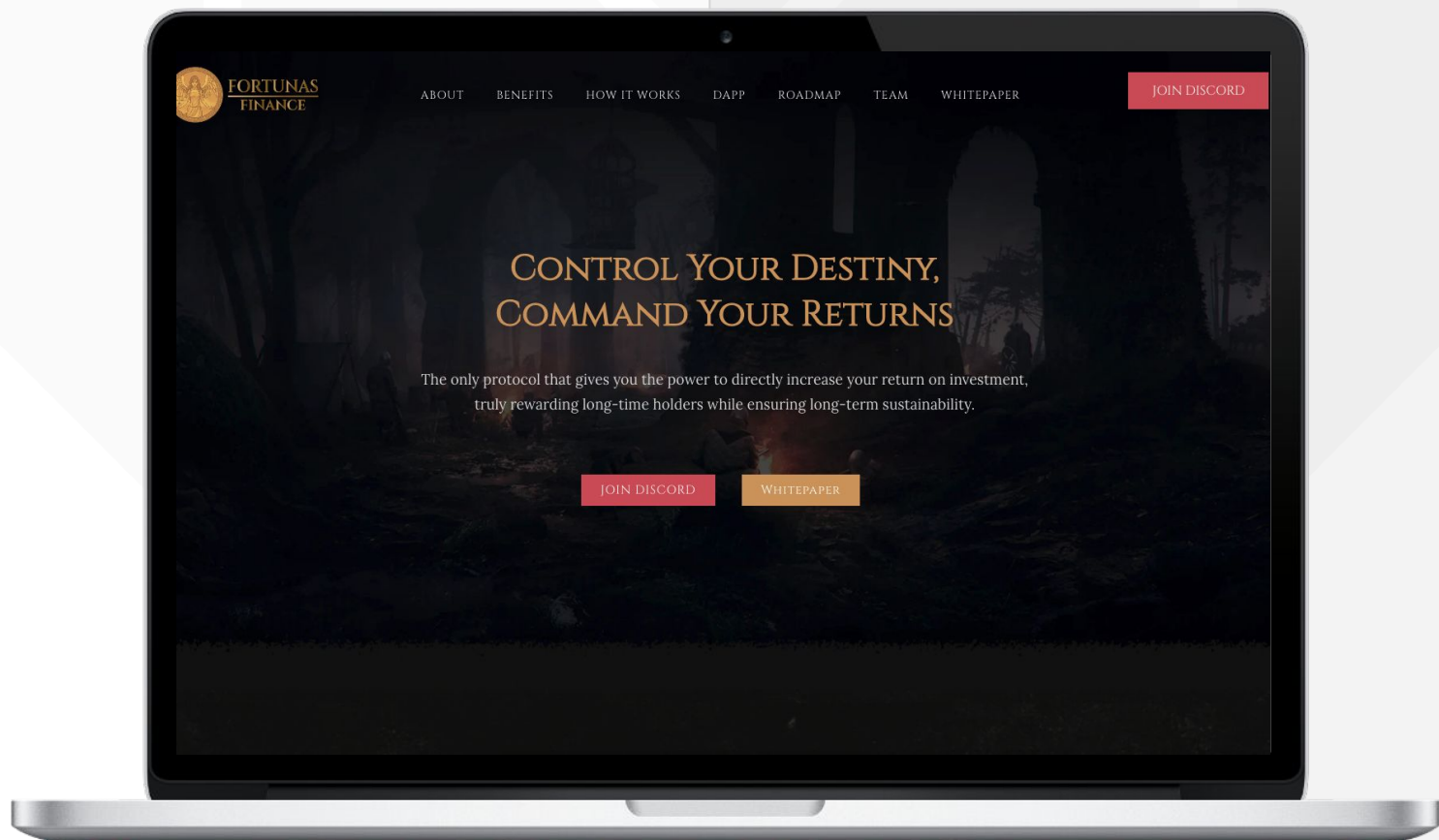
Well written, explanatory.

Roadmap

Yes, goals set with time
frames.

Mobile-friendly?

Yes



fortunas.finance

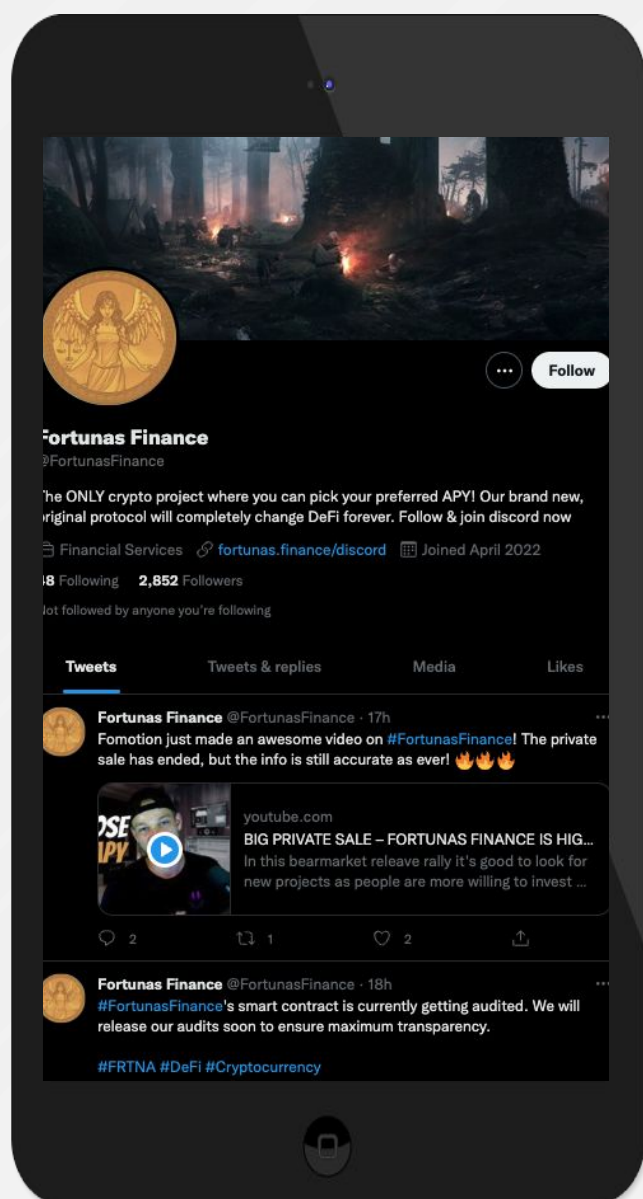


SOCIAL MEDIA & ONLINE PRESENCE



ANALYSIS

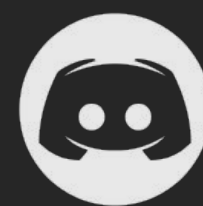
Project's social media presence is active with plenty of followers for a brand new project.



Twitter

@FortunasFinance

- 3 313 followers
- Active, few posts per day



Discord

<https://discord.com/invite/JcwbT5kugn>

- 188 members
- Active members
- Active mods



Telegram

@fortunas_fi

- 807 members
- Few active members ⚠️
- Active mods



Medium

- Not available



SPYWOLF

CRYPTO SECURITY

Audits | KYCs | dApps
Contract Development

ABOUT US

We are a growing crypto security agency offering audits, KYCs and consulting services for some of the top names in the crypto industry.

- ✓ OVER 150 SUCCESSFUL CLIENTS
- ✓ MORE THAN 500 SCAMS EXPOSED
- ✓ MILLIONS SAVED IN POTENTIAL FRAUD
- ✓ PARTNERSHIPS WITH TOP LAUNCHPADS, INFLUENCERS AND CRYPTO PROJECTS
- ✓ CONSTANTLY BUILDING TOOLS TO HELP INVESTORS DO BETTER RESEARCH

To hire us, reach out to
contact@spywolf.co or
t.me/joe_SpyWolf

FIND US ONLINE



[SPYWOLF.CO](https://spywolf.co)



[SPYWOLF.NETWORK](https://spywolf.network)



[@SPYWOLFNETWORK](https://t.me/SPYWOLFNETWORK)



[@SPYWOLFOFFICIAL](https://t.me/SPYWOLFOFFICIAL)



[@SPYWOLFNETWORK](https://twitter.com/SPYWOLFNETWORK)



[@SPYWOLFNETWORK](https://github.com/SPYWOLFNETWORK)



Disclaimer

This report shows findings based on our limited project analysis, following good industry practice from the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, overall social media and website presence and team transparency details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report.

While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the disclaimer below – please make sure to read it in full.

DISCLAIMER:

By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice.

No one shall have any right to rely on the report or its contents, and SpyWolf and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (SpyWolf) owe no duty of care towards you or any other person, nor does SpyWolf make any warranty or representation to any person on the accuracy or completeness of the report.

The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and SpyWolf hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, SpyWolf hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against SpyWolf, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report. The analysis of the security is purely based on the smart contracts, website, social media and team.

No applications were reviewed for security. No product code has been reviewed.